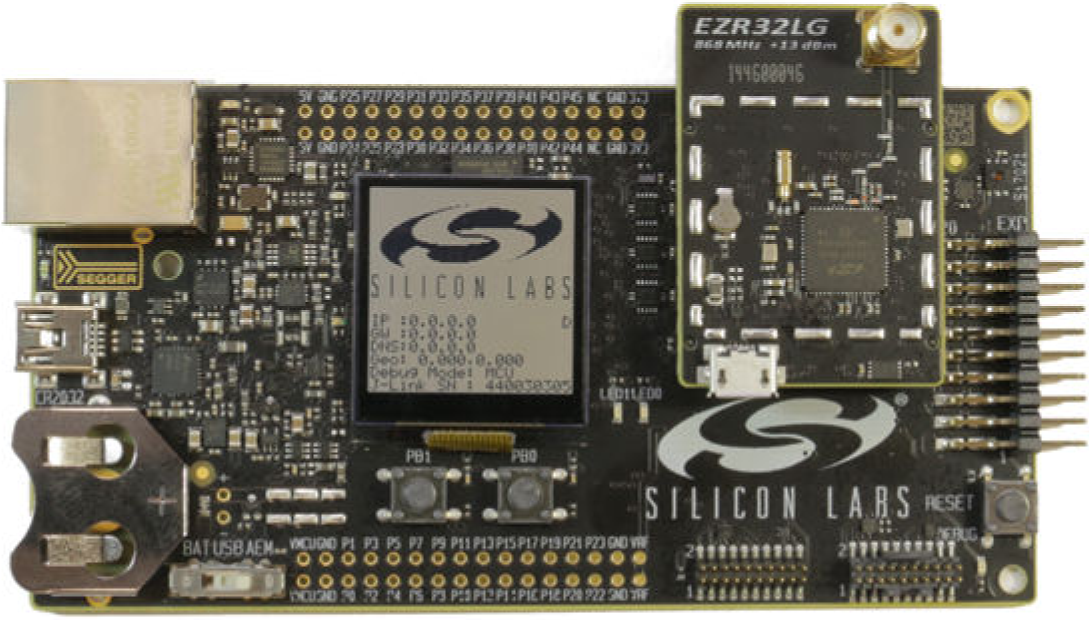
AN888: EZR32 Simple TRX Application Quick Start Guide

The EZR32LG and EZR32WG Wireless MCUs are the latest in Silicon Labs family of wireless MCUs delivering a high performance, low energy wireless solution integrated into a small form factor package. By combining a high performance sub-GHz RF transceiver with an energy efficient 32-bit MCU, the EZR32LG family provides designers the ultimate in flexibility with a family of pin-compatible devices that scale with 64/128/256 kB of flash and support Silicon Labs EZRadio or EZRadioPRO transceivers. The ultra-low power operating modes and fast wake-up times of the Silicon Labs energy friendly 32-bit MCUs, combined with the low transmit and receive power consumption of the sub-GHz radio, result in a solution optimized for battery powered applications.

The EZR32LG and EZR32WG Wireless Starter Kit (WSTK) is a low-cost, fully featured starter kit that allows designers to evaluate the performance and features of the EZR32LG/WG device family. The kit will provide all hardware and software tools needed to develop and debug applications using the EZR32LG or WG low-energy Wireless MCUs. Hardware tools include an on-board debugger, advanced energy monitoring and integrated packet trace, which are all available either over USB or Ethernet interfaces. The kit is fully supported by Silicon Labs' software libraries and the kit board support package (BSP).

KIT CONTENTS

- 2 WSTK boards
- 2 Radio cards either EZR32LG or WG for the requested frequency as listed in the doc.
- 2 USB cables to attach the WSTK to your computer.
- 2 CR2032 coin cell batteries.
- 1 card with instructions direction them to the website for Simplicity Studio and other documentation



1. Getting Started

Hardware

- BRD4001A - Wireless Starter Kit Mainboard
- One set of the following:
  - BRD4502A - EZR32LG 330 Direct Tie WSTK Radio Board; Si4460; 868 MHz +13 dBm
  - BRD4502B – EZR32LG 330 Switched WSTK Radio Board; Si4455; 434 MHz +20 dBm
  - BRD4502C - EZR32WG 330 Direct Tie WSTK Radio Board; Si4460; 868 MHz +13 dBm
  - BRD4502D – EZR32WG 330 Switched WSTK Radio Board; Si4455; 434 MHz +20 dBm
  - BRD4503A - EZR32LG 330 Switched WSTK Radio Board; Si4463; 915 MHz +20 dBm
  - BRD4503B - EZR32WG 330 Switched WSTK Radio Board; Si4463; 915 MHz +20 dBm
  - BRD4505B - EZR32WG 330 Switched WSTK Radio Board; Si4463; 490 MHz +20 dBm

Software:

- Simplicity Studio 3.0 + EFM and Wireless Packages.
- WSTK Firmware Version 2.4 or later. (Update available within Simplicity Studio through the Kit Manager)

Software Dependencies

- Simplicity Studio v3

Installing All Required Software

Install all of the above listed software. All of the EZR32 development kit software and prerequisite software must be installed on your machine for the development kit software to work correctly. The order in which the software is installed is not important but you will want to make sure that you have it all installed before you launch Simplicity Studio and begin working on your first project.

Connecting Your Hardware

Once you have installed all the required software you can connect your EZR32LG/WG development kit hardware to your PC. It is simply connected using a USB cable as shown below:

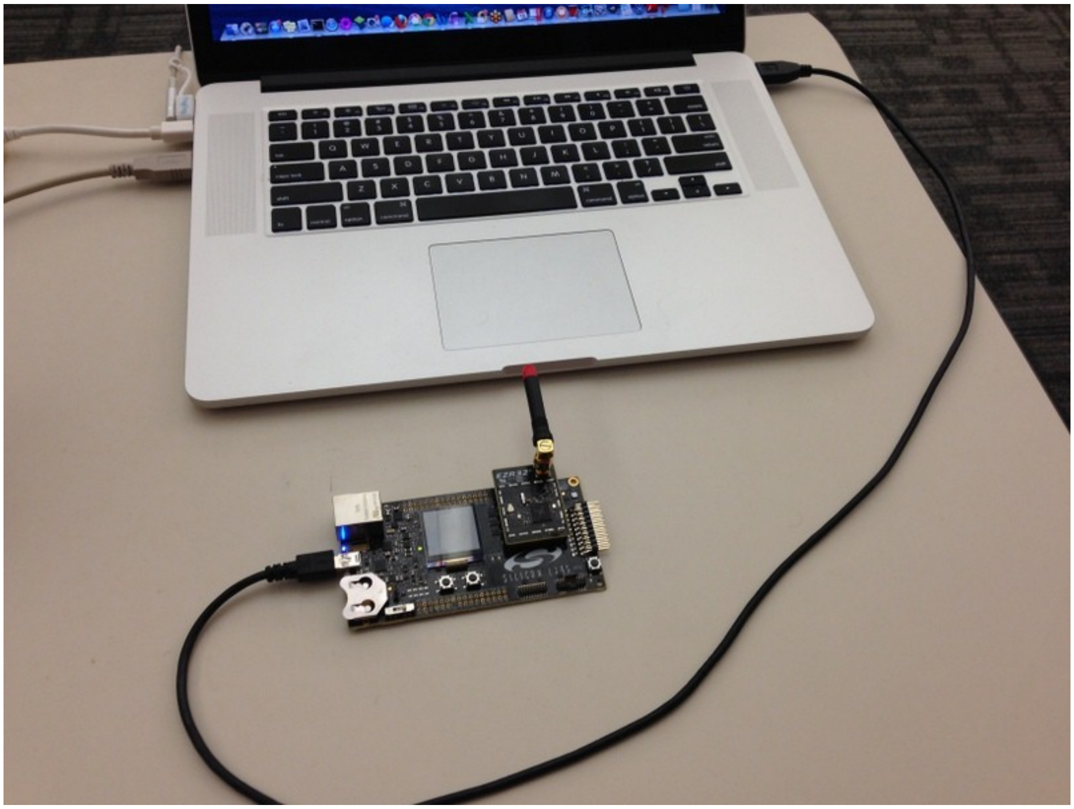


Figure 1.1. WSTK Board Connected to PC with USB

Note: For best performance in Simplicity Studio please be sure that the power switch on your WSTK is in the Advanced Energy Monitoring or “AEM” position.

2. Running Simplicity Studio for the First Time

Navigation in Simplicity Studio

Simplicity Studio is built on the Eclipse platform. As such, it is broken up into different “perspectives” each of which allow access to a specific set of functionality. Simplicity Studio starts up in the “Simplicity Perspective,” sometimes referred to as the “Home Screen.”

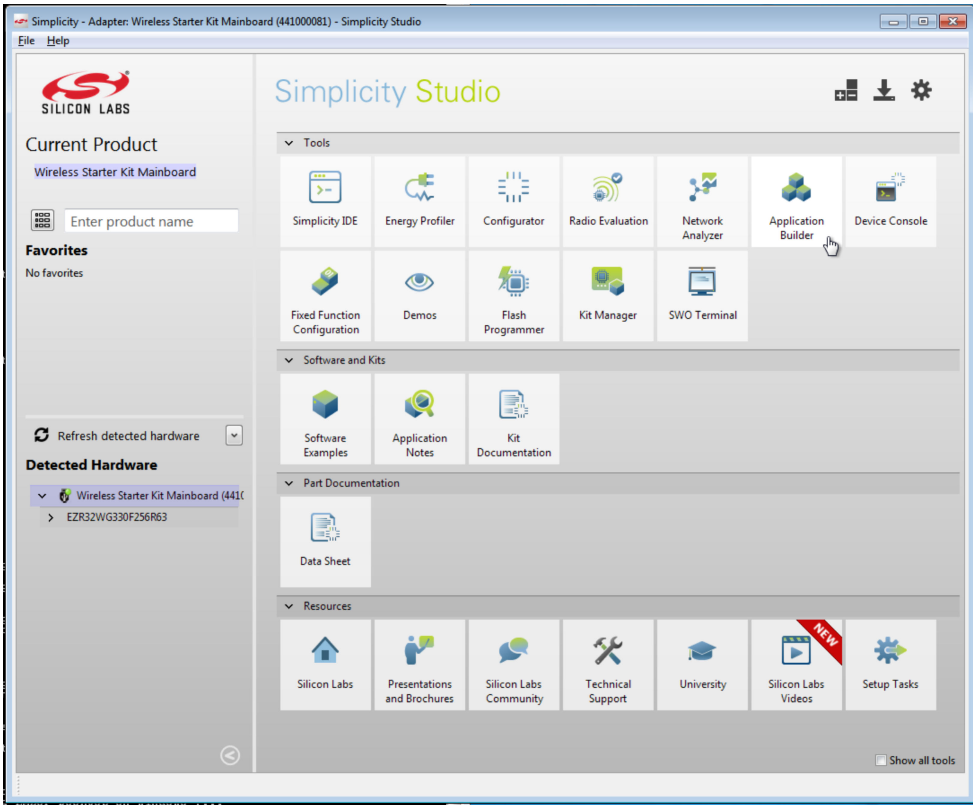
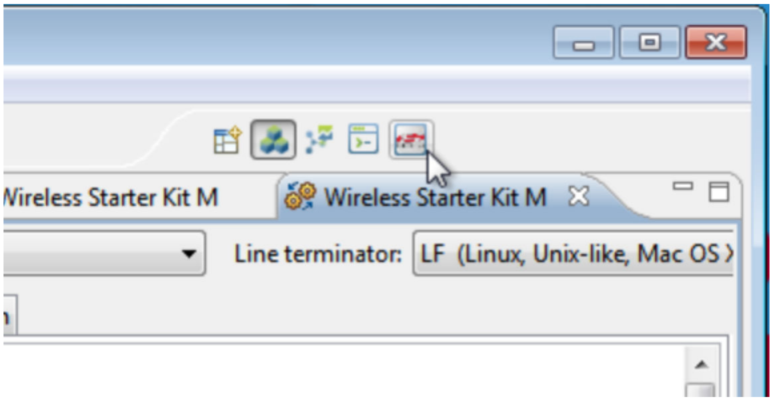


Figure 2.1. Simplicity Studio's Simplicity Perspective

From the Simplicity Perspective, you can discover devices, configure Simplicity Studio, or navigate to another perspective for application development.

The Simplicity Perspective shows large tile icons that represent the various pieces of functionality within Simplicity Studio. When you are in a different perspective you will see smaller tile icons in the top right-hand corner of your screen. You can always navigate back to the Simplicity Perspective or any other perspective at any time by clicking on one of the tile icons in the top right-hand corner of your screen.



Line terminator: LF (Linux, Unix-like, Mac OS

Figure 2.2. Navigation Tile Icons

Downloading Packages for Simplicity Studio

By default Simplicity Studio comes with some core components necessary for development. The real substance of the Simplicity Studio application is provided in the “Packages” available from the Simplicity Studio update site. Before you begin working with Simplicity Studio, you will want to make sure that you have all of the packages installed that you need for your development platform. In particular, for the EZR32 family of products, you will need both the “Wireless” and “EFM Product” packages.

To download packages for Simplicity Studio, [**click**] on the Package Management icon in the top right-hand corner of the Simplicity Perspective.

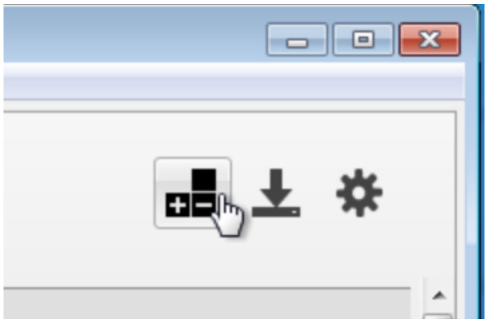
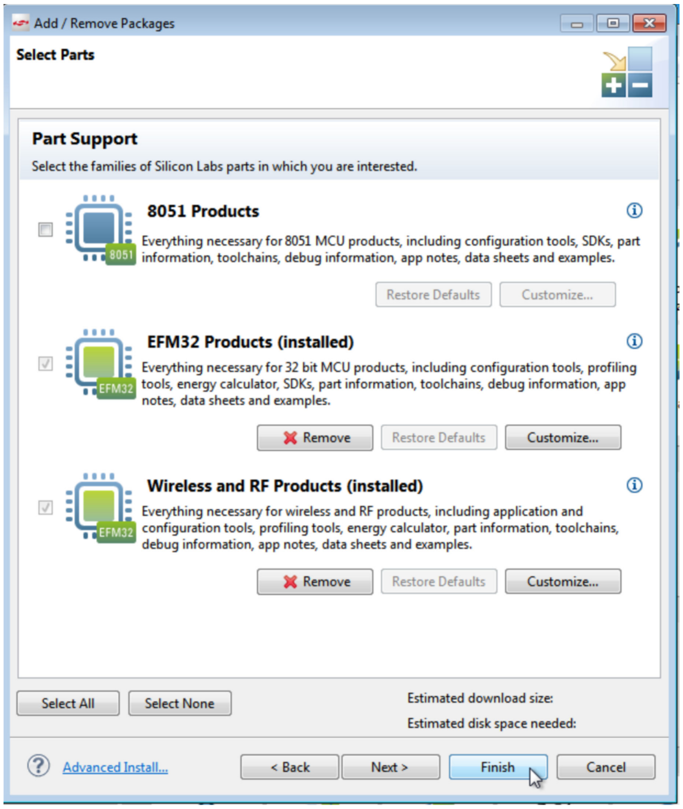


Figure 2.3. Packages Management

The Add/Remove Packages window opens and will guide you through the process of updating Simplicity Studio’s packages. You will need both the wireless package and the EFM SDK package for development on the EZR32 product.



Add / Remove Packages

Select Parts

Part Support

Select the families of Silicon Labs parts in which you are interested.

□ 8051 Products

Everything necessary for 8051 MCU products, including configuration tools, SDKs, part information, toolchains, debug information, app notes, data sheets and examples.

☑ EFM32 Products (installed)

Everything necessary for 32 bit MCU products, including configuration tools, profiling tools, energy calculator, SDKs, part information, toolchains, debug information, app notes, data sheets and examples.

☑ Wireless and RF Products (installed)

Everything necessary for wireless and RF products, including application and configuration tools, profiling tools, energy calculator, part information, toolchains, debug information, app notes, data sheets and examples.

Estimated download size:

Estimated disk space needed:

Figure 2.4. Add/Remove Packages Window

Updating Simplicity Studio

In addition to downloading packages, you can also update the Simplicity Studio core software at any time by clicking on the [**Update Software**] icon in the top right-hand corner of the Simplicity Perspective.

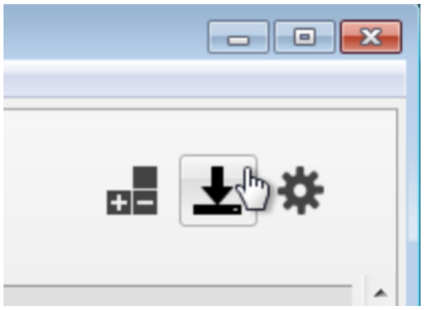
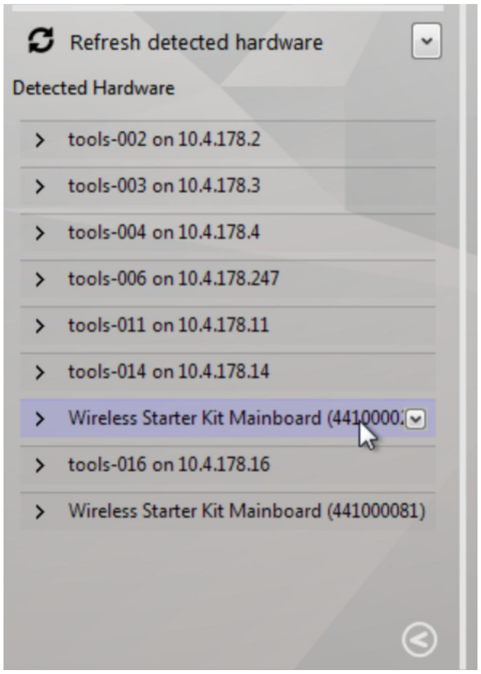


Figure 2.5. Updating Simplicity Studio

Discovering Your Development Kit Hardware in Simplicity Studio

When you launch Simplicity Studio, it should automatically discover your USB connected Wireless Starter Kit (WSTK) hardware and display it in the Simplicity Perspective as shown below:



Refresh detected hardware

Detected Hardware

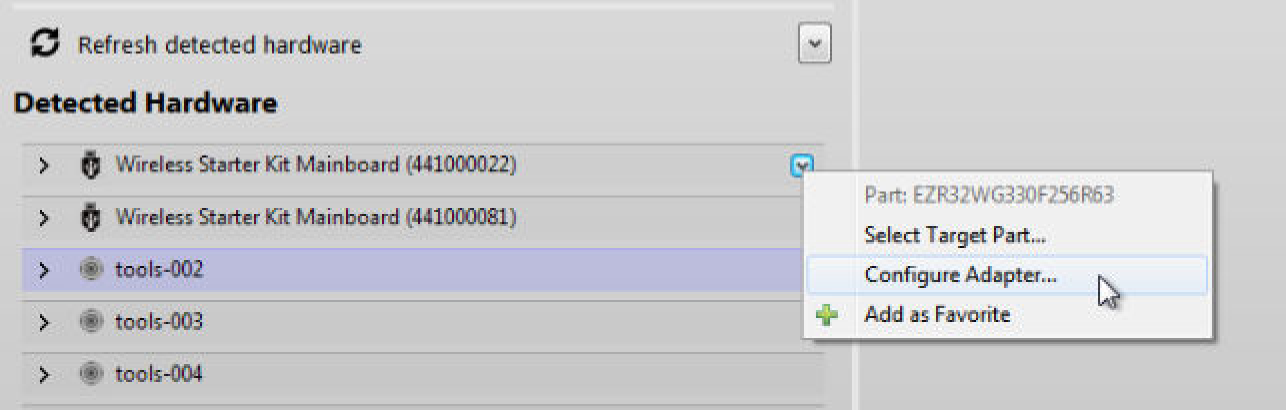
- tools-002 on 10.4.178.2
- tools-003 on 10.4.178.3
- tools-004 on 10.4.178.4
- tools-006 on 10.4.178.247
- tools-011 on 10.4.178.11
- tools-014 on 10.4.178.14
- Wireless Starter Kit Mainboard (4410000
- tools-016 on 10.4.178.16
- Wireless Starter Kit Mainboard (441000081)

Figure 2.6. Detected Hardware

The WSTK board will come up with an identifier that includes the serial number of the WSTK connected. The serial number displayed in Simplicity Studio should match the number displayed on the WSTK board’s LCD.

Configuring Your WSTK for an Ethernet Connection

You can also connect to your WSTK boards to be connect and discovered via Ethernet. You can configure the IP address of your device through the Simplicity Studio Home Screen by clicking on the arrow to the right of your device and selecting [**Configure Adapter**].

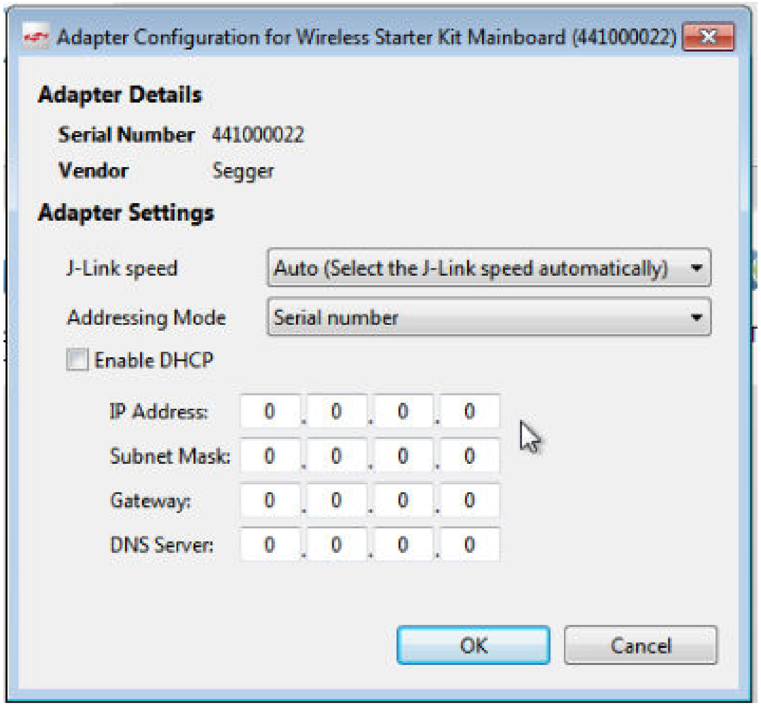


Detected Hardware

- Wireless Starter Kit Mainboard (441000022)
- Wireless Starter Kit Mainboard (441000081)
- tools-002
- tools-003
- tools-004

Figure 2.7. Select Configure Adapter

The Configure Adapter dialog window will allow you to set the J-Link Speed, Addressing Mode, and IP configuration for your WSTK board.



Adapter Configuration for Wireless Starter Kit Mainboard (441000022)

Adapter Details

Serial Number	441000022
Vendor	Segger

Adapter Settings

J-Link speed: Auto (Select the J-Link speed automatically)

Addressing Mode: Serial number

□ Enable DHCP

IP Address:	0	0	0	0
Subnet Mask:	0	0	0	0
Gateway:	0	0	0	0
DNS Server:	0	0	0	0

Figure 2.8. Configure Adapter Dialog Window

Discovering your WSTKs via Ethernet

In order to discover your WSTKs via the Ethernet connection you must first configure Simplicity Studio to search for and discover your devices. You can do this by clicking on the arrow at the top of the Adapters View in the Simplicity Perspective and selecting [**Configure Device Scanning**].

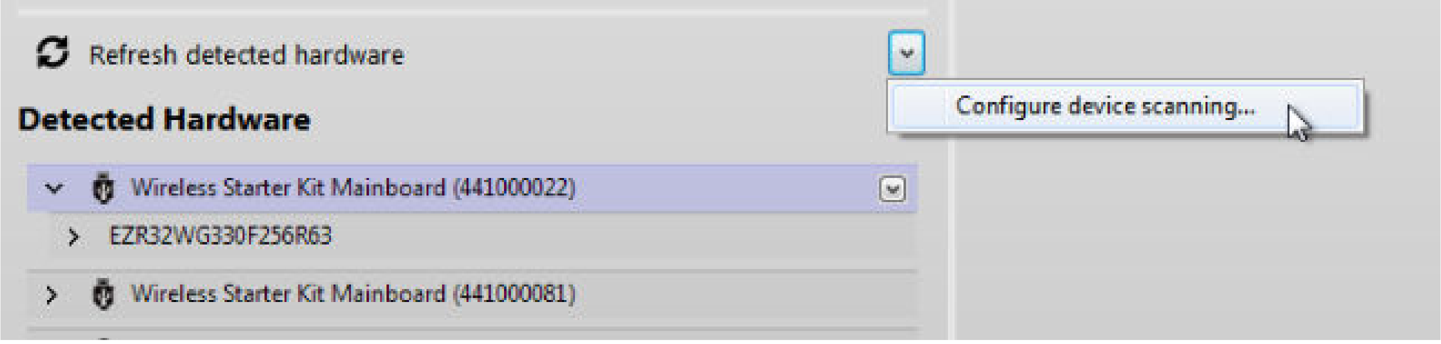
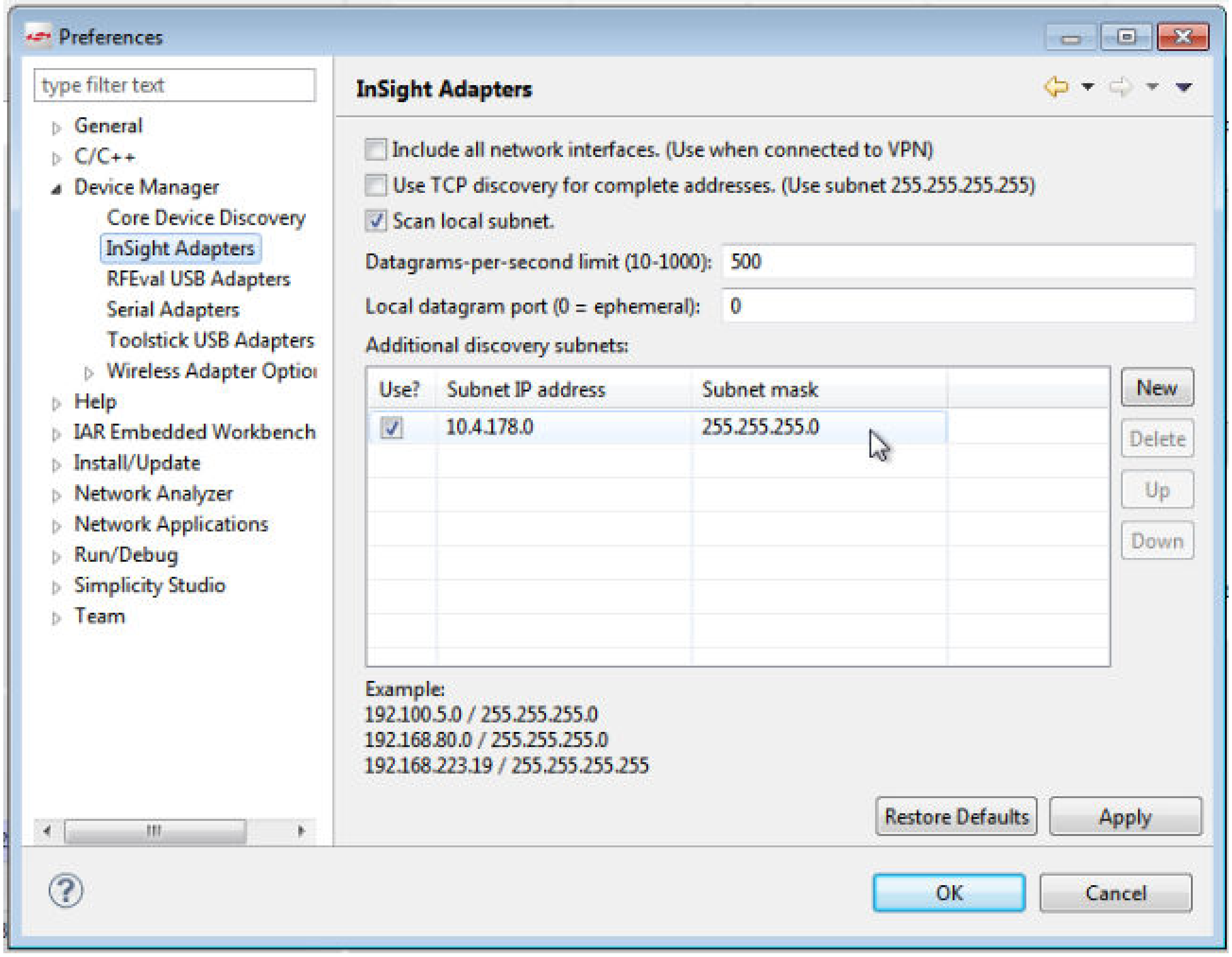

Refresh detected hardware
Detected Hardware
Wireless Starter Kit Mainboard (441000022)
EZR32WG330F256R63
Wireless Starter Kit Mainboard (441000081)
Configure device scanning...

Figure 2.9. Configure Device Scanning



InSight Adapters

□ Include all network interfaces. (Use when connected to VPN)

□ Use TCP discovery for complete addresses. (Use subnet 255.255.255.255)

☑ Scan local subnet.

Datagrams-per-second limit (10-1000): 500

Local datagram port (0 = ephemeral): 0

Additional discovery subnets:

Use?	Subnet IP address	Subnet mask
☑	10.4.178.0	255.255.255.0

Example:
192.100.5.0 / 255.255.255.0
192.168.80.0 / 255.255.255.0
192.168.223.19 / 255.255.255.255

Figure 2.10. InSight Adapters

The Configuration interface located in [**Preferences > Device Manger > InSight Adapters**] allows you to configure the scanning that Simplicity Studio will do to discover adapters on the network. In the case above, there is a WSTK connected with an IP address of 10.4.178.121. In order to scan for all devices on that subnet of the test network, set the subnet IP address at 10.4.178.0 with a mask of 255.255.255.0. This will scan for all WSTKs on the subnet 10.4.178 and should return my device.

Upgrading the WSTK Boards to the Latest Firmware

Once you have connected the WSTK boards to the computer, make sure that the boards have the latest WSTK firmware loaded. This is important in order to ensure that they will communicate with the Simplicity Studio software correctly.

Simplicity Studio includes a bundled Kit Manager that can be used to upgrade your WSTK boards. To access the Kit Manager within Simplicity Studio, click on the WSTK board that needs to be upgraded so that it shows as selected, and then click on the [**Kit Manager**] Tile on the right-hand side of the Simplicity Perspective.

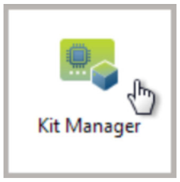
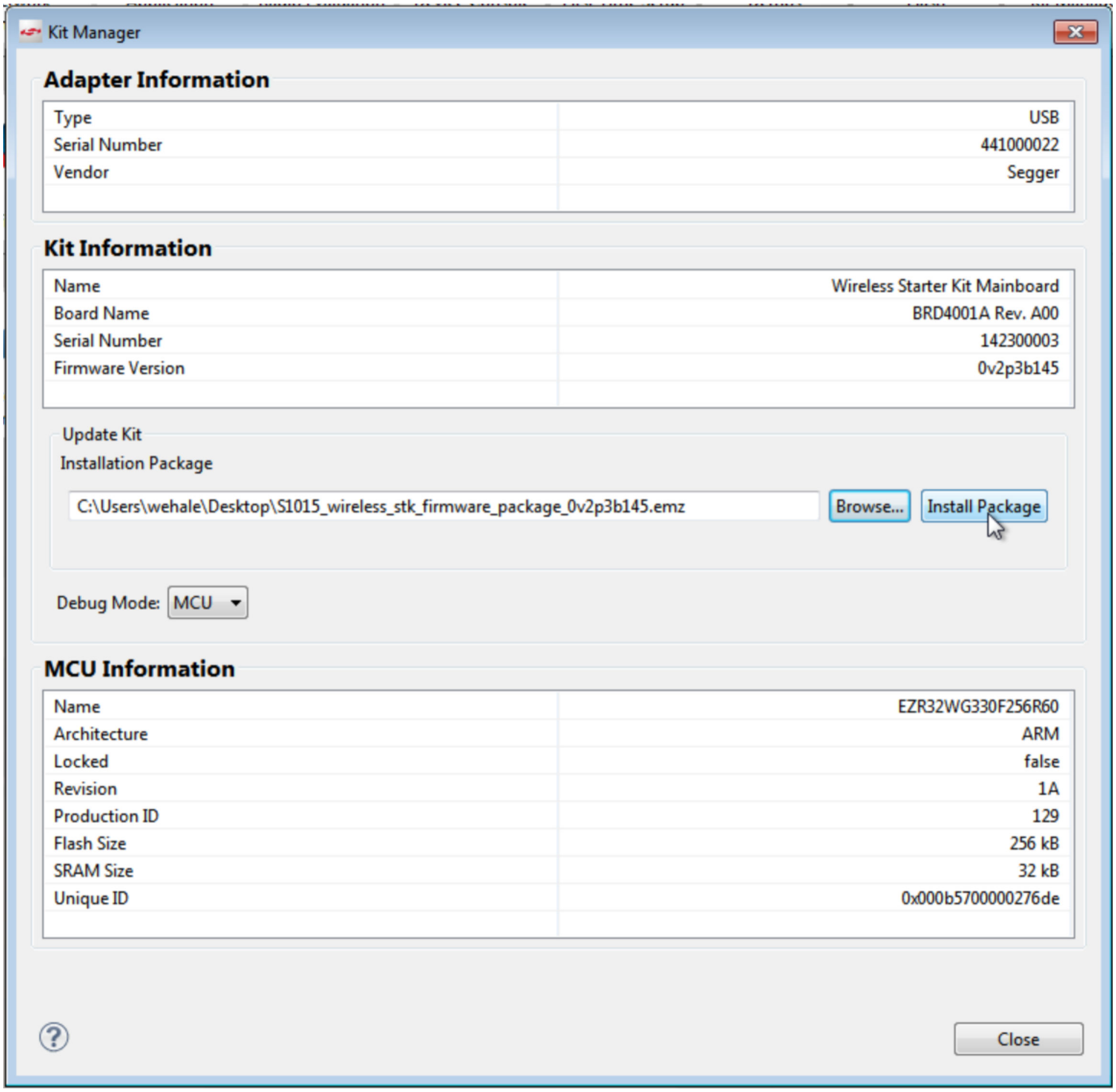


Figure 2.11. Kit Manager

Clicking on the Kit Manager tile will open the Kit Manager dialog window, through which you can load a new firmware image onto your WSTK main board. The Kit Manager has a bundled version of the latest WSTK software that it will ask to load onto your device.



Adapter Information

Type	USB
Serial Number	441000022
Vendor	Segger

Kit Information

Name	Wireless Starter Kit Mainboard
Board Name	BRD4001A Rev. A00
Serial Number	142300003
Firmware Version	0v2p3b145

Update Kit

Installation Package

C:\Users\wehale\Desktop\S1015_wireless_stk_firmware_package_0v2p3b145.emz

Debug Mode: MCU

MCU Information

Name	EZR32WG330F256R60
Architecture	ARM
Locked	false
Revision	1A
Production ID	129
Flash Size	256 kB
SRAM Size	32 kB
Unique ID	0x000b5700000276de

Figure 2.12. Kit Manager Dialog

Configuring Your Preferred IDE

Simplicity Studio is an Integrated Development Environment (IDE). However, you are not limited to using only Simplicity Studio as your IDE, you have the option of configuring Simplicity Studio to launch your preferred IDE for the development of software samples. For instance, if you would prefer to use IAR or Keil, you can configure this option within Simplicity Studio by clicking on the “Settings” icon in the top right corner of the Simplicity Perspective and going to [**Simplicity Studio > Preferred IDE**].

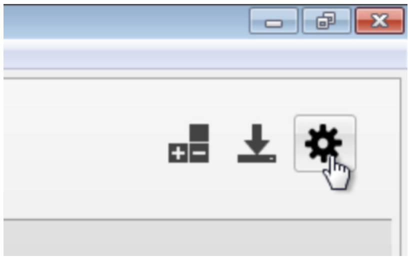
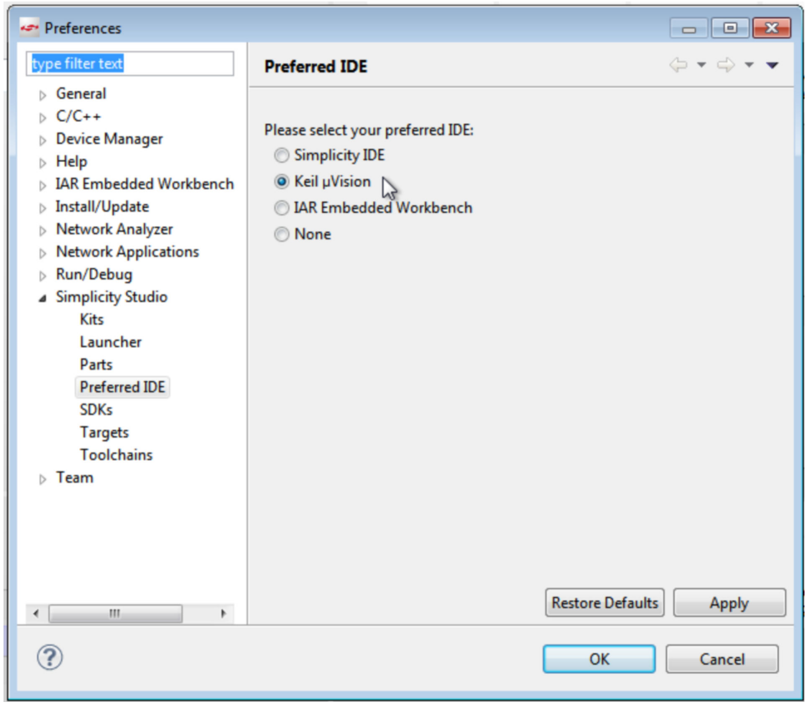


Figure 2.13. Settings Icon



Please select your preferred IDE:

○ Simplicity IDE
● Keil µVision
○ IAR Embedded Workbench
○ None

Figure 2.14. Preferences/Preferred IDE

The “Preferred IDE” will be used when launching sample applications from within Simplicity Studio.
Note: IAR and GCC are supported as compiler toolchains within Simplicity Studio, meaning that the IAR or GCC compiler can be used within the Simplicity Studio IDE. Keil and other compilers are only supported as “Preferred IDEs,” meaning that a project file is provided in the sample applications but the Keil compiler is not integrated into Simplicity Studio. So if you want to use the Keil compiler, you must use it within the Keil uVision IDE.

3. RF Evaluation Using Nodetest

Testing the EZR32 Using the "Nodetest" Application

The “nodetest” application provides a Command Line Interface (CLI) to Simplicity Studio for RF Evaluation of a device. In order to load the nodetest application, do the following:

1. In the Simplicity Studio Perspective, select one of the RF devices you would like to load nodetest onto.
2. With the device selected, select the [**Network Analyzer**] tile on the right-hand side.

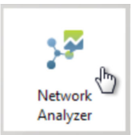


Figure 3.1. Network Analyser Tile

3. The Network Analyzer Perspective opens. In this perspective you will see the Adapters View on the left-hand side of the screen.
4. Rightclick on the device to which you would like to load the nodetest application, and select [**Connect**] to connect to the device.
5. Right-click on the WSTK board in the Adapter’s menu again and select [**Flash / Upload**].

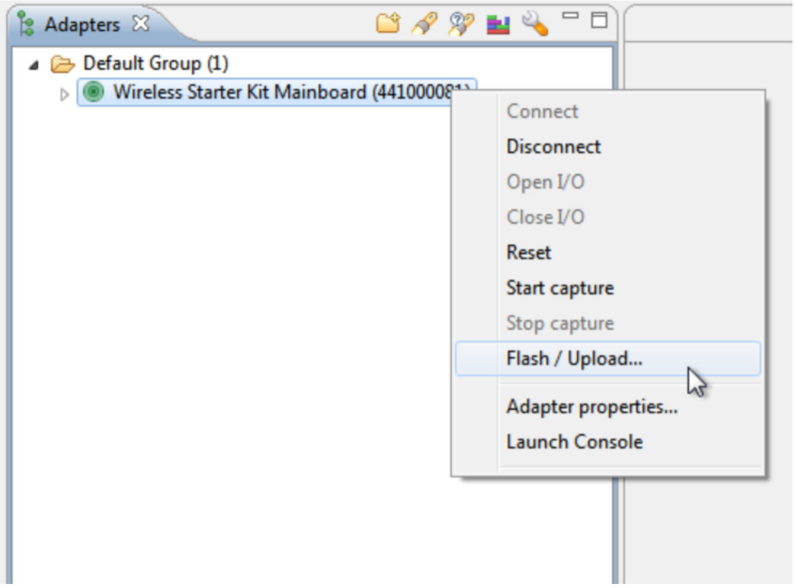
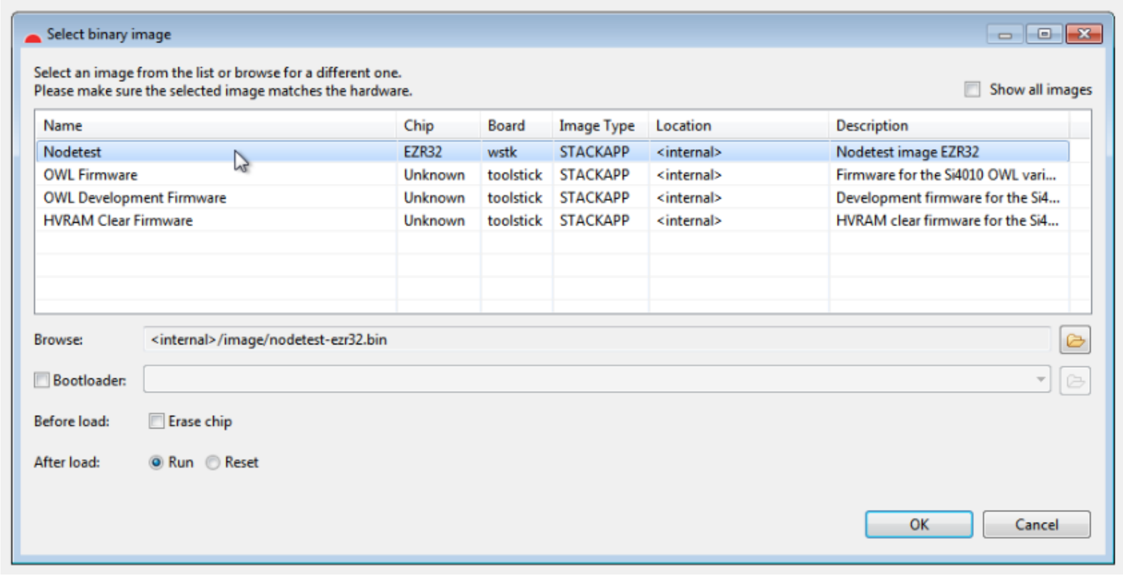


Figure 3.2. Flash/Upload

6. Choose from the list of applications the nodetest application that is appropriate for your device and click [**OK**]. You will notice that there several nodetest images available depending on your frequency band: 900 covering 915 and 800 covering 868, 490, and 434. Be sure to select the image that supports the radio card on your WSTK board.



Select an image from the list or browse for a different one.
Please make sure the selected image matches the hardware.

□ Show all images

Name	Chip	Board	Image Type	Location	Description
Nodetest	EZR32	wstk	STACKAPP	<internal>	Nodetest image EZR32
OWL Firmware	Unknown	toolstick	STACKAPP	<internal>	Firmware for the Si4010 OWL vari...
OWL Development Firmware	Unknown	toolstick	STACKAPP	<internal>	Development firmware for the Si4...
HVRAM Clear Firmware	Unknown	toolstick	STACKAPP	<internal>	HVRAM clear firmware for the Si4...

Browse: <internal>/image/nodetest-ezr32.bin

□ Bootloader:

Before load: □ Erase chip

After load: ● Run ○ Reset

Figure 3.3. Choose Nodetest

7. The nodetest application will load onto your EZR32LG or WG wireless MCU. You will see a progress indicator on the adapter indicating that the device is being loaded. Once the application load is complete, the name of the application loaded (nodetest) will appear to the right of the adapter.

Running Console for Command Line Interface to Nodetest Application

Once the Nodetest application is loaded onto the EZR32LG/WG, you must first [**Connect**] to the device through the Adapters view, and then you can interact with the nodetest application using the Command Line Interface within the Console view in Simplicity Studio.

Right-click on the [**Adapter**] in the Adapter's View and select [**Connect**] and then [**Launch Console**].

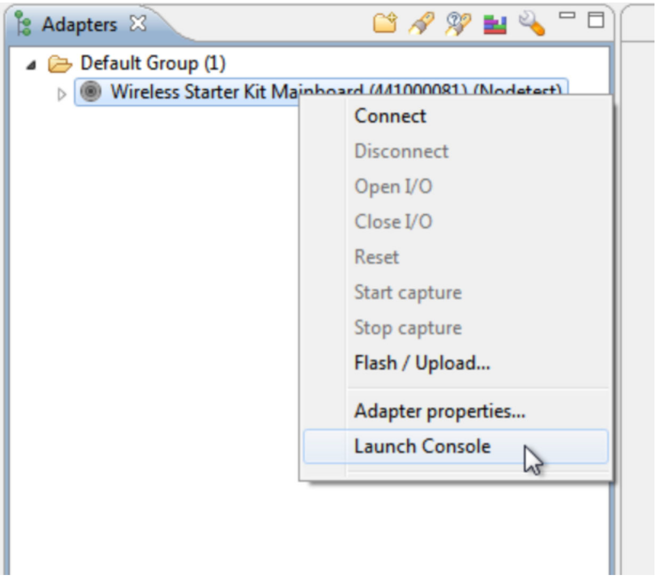
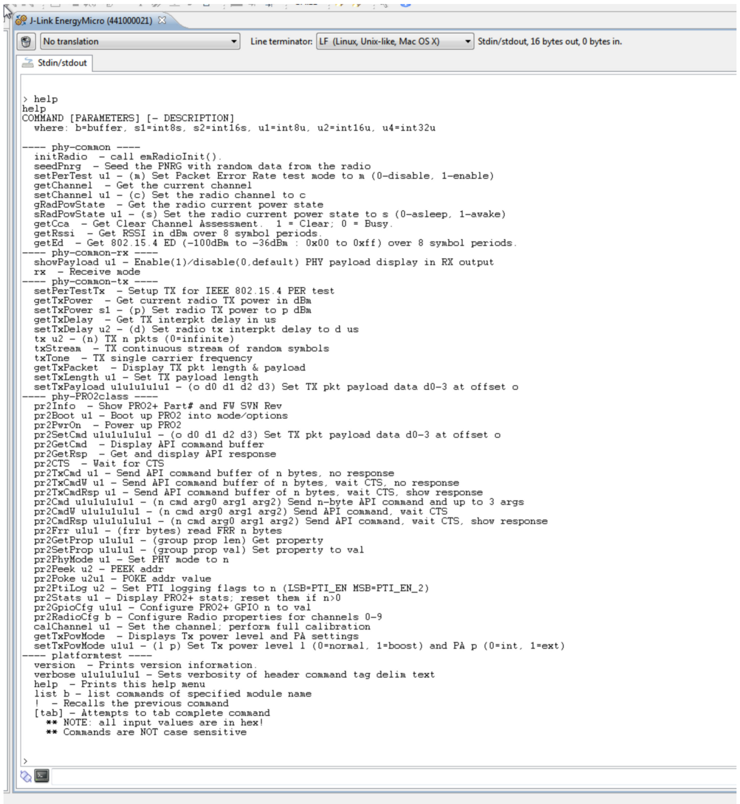


Figure 3.4. Launch Console

The Console View will give you a CLI interface in Simplicity Studio's Network Analyzer Perspective so that you can interact directly with the nodetest application.



No translation   Line terminator: LF (Linux, Unix-like, Mac OS X)   Stdin/stdout, 16 bytes out, 0 bytes in.

```
> help
help
COMMAND [PARAMETERS] [- DESCRIPTION]
  where: b=buffer, s1=int8s, s2=int16s, u1=int8u, u2=int16u, u4=int32u

---- phy-common ----
  initRadio  - call emRadioInit().
  seedPnrg  - Seed the PNRG with random data from the radio
  setPerTest u1 - (m) Set Packet Error Rate test mode to m (0-disable, 1-enable)
  getChannel  - Get the current channel
  setChannel u1 - (c) Set the radio channel to c
  gRadPowState  - Get the radio current power state
  sRadPowState u1 - (s) Set the radio current power state to s (0-asleep, 1-awake)
  getCca  - Get Clear Channel Assessment.  1 = Clear; 0 = Busy.
  getRssi  - Get RSSI in dBm over 8 symbol periods.
  getEd  - Get 802.15.4 ED (-100dBm to -36dBm : 0x00 to 0xff) over 8 symbol periods.
---- phy-common-rx ----
  showPayload u1 - Enable(1)/disable(0,default) PHY payload display in RX output
  rx  - Receive mode
---- phy-common-tx ----
  setPerTestTx  - Setup TX for IEEE 802.15.4 PER test
  getTxPower  - Get current radio TX power in dBm
  setTxPower s1 - (p) Set radio TX power to p dBm
  getTxDelay  - Get TX interpkt delay in us
  setTxDelay u2 - (d) Set radio tx interpkt delay to d us
  tx u2 - (n) TX n pkts (0=infinite)
  txStream  - TX continuous stream of random symbols
  txTone  - TX single carrier frequency
  getTxPacket  - Display TX pkt length & payload
  setTxLength u1 - Set TX payload length
  setTxPayload u1u1u1u1u1 - (o d0 d1 d2 d3) Set TX pkt payload data d0-3 at offset o
---- phy-PRO2class ----
  pr2Info  - Show PRO2+ Part# and FW SVN Rev
  pr2Boot u1 - Boot up PRO2 into mode/options
  pr2PwrOn  - Power up PRO2
  pr2SetCmd u1u1u1u1u1 - (o d0 d1 d2 d3) Set TX pkt payload data d0-3 at offset o
  pr2GetCmd  - Display API command buffer
  pr2GetRsp  - Get and display API response
  pr2CTS  - Wait for CTS
  pr2TxCmd u1 - Send API command buffer of n bytes, no response
  pr2TxCmdW u1 - Send API command buffer of n bytes, wait CTS, no response
  pr2TxCmdRsp u1 - Send API command buffer of n bytes, wait CTS, show response
  pr2Cmd u1u1u1u1u1 - (n cmd arg0 arg1 arg2) Send n-byte API command and up to 3 args
  pr2CmdW u1u1u1u1u1 - (n cmd arg0 arg1 arg2) Send API command, wait CTS
  pr2CmdRsp u1u1u1u1u1 - (n cmd arg0 arg1 arg2) Send API command, wait CTS, show response
  pr2Frr u1u1 - (frr bytes) read FRR n bytes
  pr2GetProp u1u1u1 - (group prop len) Get property
  pr2SetProp u1u1u1 - (group prop val) Set property to val
  pr2PhyMode u1 - Set PHY mode to n
  pr2Peek u2 - PEEK addr
  pr2Poke u2u1 - POKE addr value
  pr2PtiLog u2 - Set PTI logging flags to n (LSB=PTI_EN MSB=PTI_EN_2)
  pr2Stats u1 - Display PRO2+ stats; reset them if n>0
  pr2GpioCfg u1u1 - Configure PRO2+ GPIO n to val
  pr2RadioCfg b - Configure Radio properties for channels 0-9
  calChannel u1 - Set the channel; perform full calibration
  getTxPowMode  - Displays Tx power level and PA settings
  setTxPowMode u1u1 - (l p) Set Tx power level l (0=normal, 1=boost) and PA p (0=int, 1=ext)
---- platformtest ----
  version  - Prints version information.
  verbose u1u1u1u1u1 - Sets verbosity of header command tag delim text
  help  - Prints this help menu
  list b - list commands of specified module name
  !  - Recalls the previous command
  [tab] - Attempts to tab complete command
    ** NOTE: all input values are in hex!
    ** Commands are NOT case sensitive


>
```

Figure 3.5. Nodetest CLI: Help

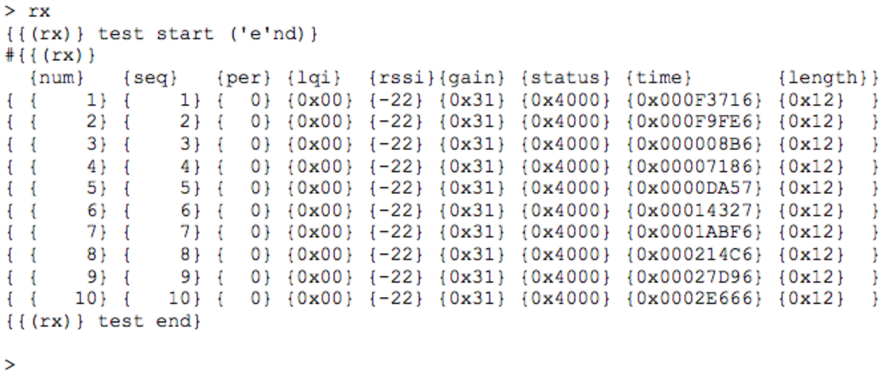
Typing `help` on the command line gives you the complete list of commands available in the nodetest application.

Running a Packet Error Test from the CLI

A typical usage scenario for the nodetest application is the measurement of Packet Error Rate (PER). PER is defined in 802.15.4 as the percentage of transmitted packets that are not detected correctly. The Nodetest application provides two commands `rx` and `tx` that are designed to interoperate with each other across two nodes running this application.

PER measurement procedure:

1. Load the nodetest application on two EZR32LG nodes.
2. Set the radio to the desired channel on both nodes with the `setchannel n` command where "n" is the desired channel number in hex.
3. Issue the `rx` command on the first node.
4. Issue the `tx <hex value>` command on the second node. (This command will transmit 10 (hex a) packets of the form required for PER analysis. For actual PER analysis, send 1000 or more packets.)
5. Observe the PER result in the third column of the `rx` command output, for example:



```
> rx
{{(rx)} test start ('e'nd)}
#{{(rx)}
  {num}    {seq}    {per}  {lqi}  {rssi}{gain} {status} {time}       {length}}
{ {     1} {     1} {   0} {0x00} {-22} {0x31} {0x4000} {0x000F3716} {0x12}   }
{ {     2} {     2} {   0} {0x00} {-22} {0x31} {0x4000} {0x000F9FE6} {0x12}   }
{ {     3} {     3} {   0} {0x00} {-22} {0x31} {0x4000} {0x000008B6} {0x12}   }
{ {     4} {     4} {   0} {0x00} {-22} {0x31} {0x4000} {0x00007186} {0x12}   }
{ {     5} {     5} {   0} {0x00} {-22} {0x31} {0x4000} {0x0000DA57} {0x12}   }
{ {     6} {     6} {   0} {0x00} {-22} {0x31} {0x4000} {0x00014327} {0x12}   }
{ {     7} {     7} {   0} {0x00} {-22} {0x31} {0x4000} {0x0001ABF6} {0x12}   }
{ {     8} {     8} {   0} {0x00} {-22} {0x31} {0x4000} {0x000214C6} {0x12}   }
{ {     9} {     9} {   0} {0x00} {-22} {0x31} {0x4000} {0x00027D96} {0x12}   }
{ {    10} {    10} {   0} {0x00} {-22} {0x31} {0x4000} {0x0002E666} {0x12}   }
{{(rx)} test end}

>
```

Figure 3.6. PER Measurement

Running the TX Test from the Simplicity Studio GUI

The Simplicity Studio RF Evaluation interface allows you to configure the radio prior to running a test via a Graphical User Interface (GUI). To run the TX portion of your test using the GUI in Simplicity Studio you may either click on the [**Radio Evaluation**] Tile on the Simplicity Studio Home Screen or go to [**Window > Show View > Other… >InSight RF Evaluation > Tests**].

Click on the [**Radio Evaluation**] Tile in the Simplicity Perspective to launch the Radio Evaluation tools.

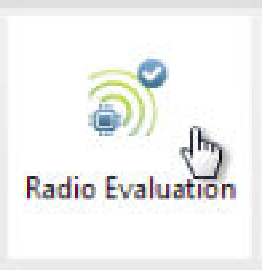
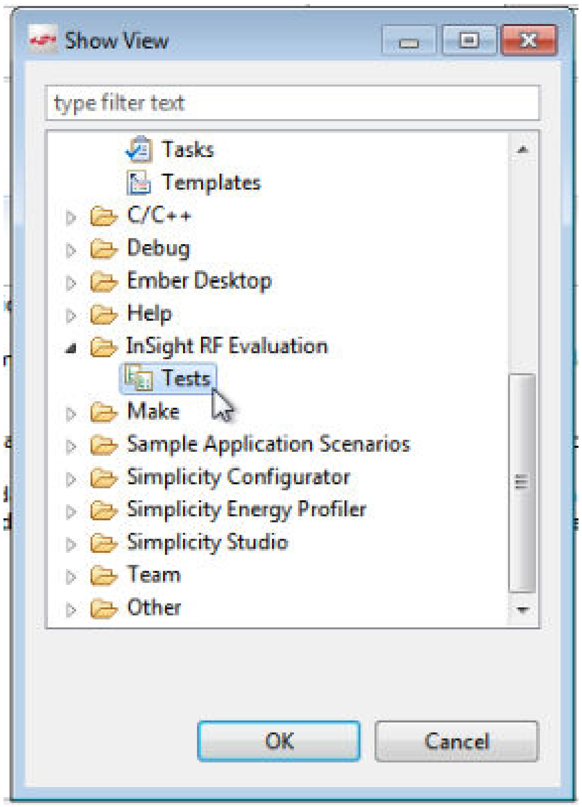


Figure 3.7. Radio Evaluation Tile

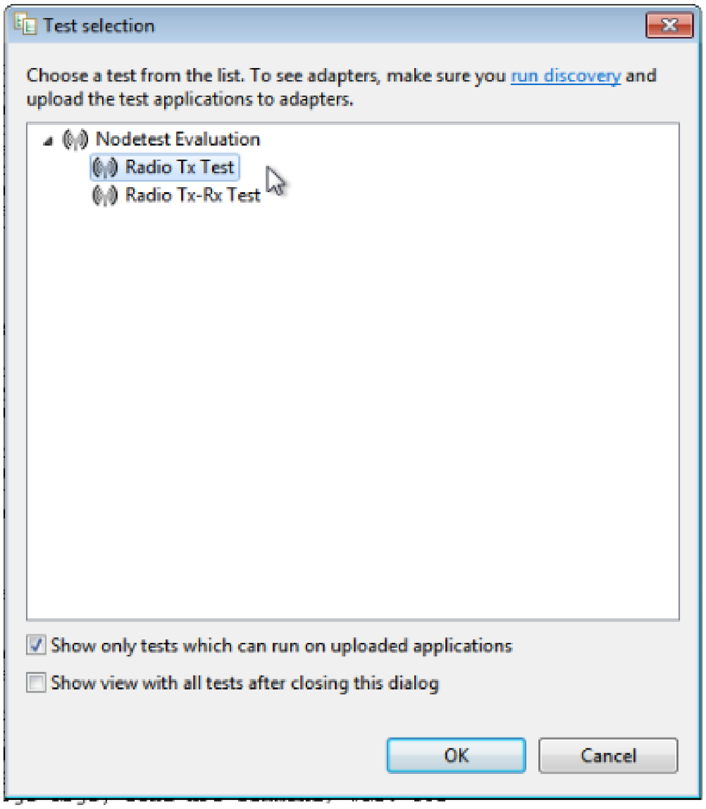

Show View

type filter text

- Tasks
- Templates
- C/C++
- Debug
- Ember Desktop
- Help
- InSight RF Evaluation
  - Tests
- Make
- Sample Application Scenarios
- Simplicity Configurator
- Simplicity Energy Profiler
- Simplicity Studio
- Team
- Other

Figure 3.8. Radio Evaluation Test View

The nodetest application must be loaded onto your device before you can run and interact with the Radio Evaluation interface in Simplicity Studio. Either of the above access methods will pop up a dialog in which you can select the radio test you wish to run. Once you select the test, click [**OK**].



Test selection

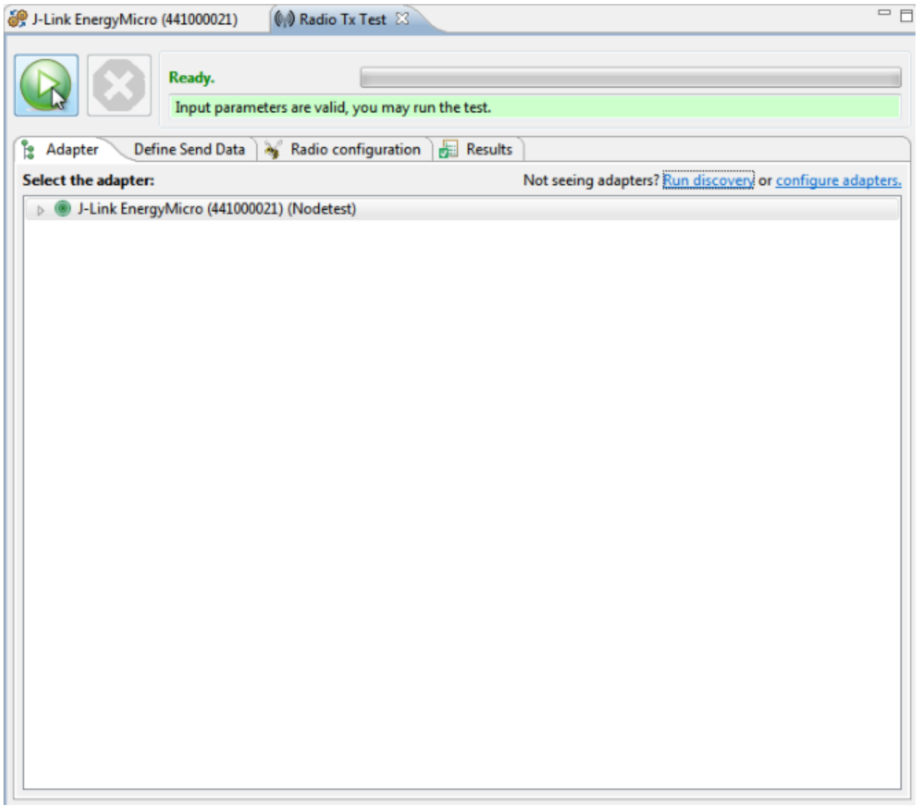
Choose a test from the list. To see adapters, make sure you run discovery and upload the test applications to adapters.

☑ Show only tests which can run on uploaded applications

□ Show view with all tests after closing this dialog

Figure 3.9. Radio TX Test Selection

This will launch a Radio Tx Test View from which you can configure and run your Tx Test. From the Test View, you may run your test at any time by clicking on the [**green arrow**].



Ready.

Input parameters are valid, you may run the test.

Select the adapter:

Not seeing adapters? Run discovery or configure adapters.

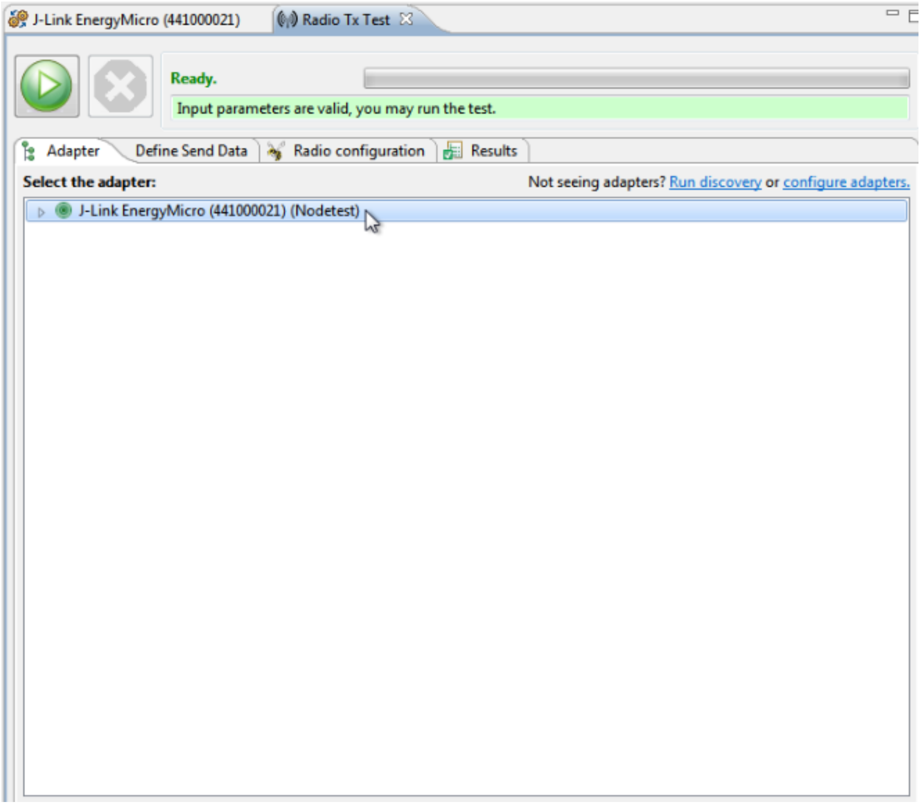
J-Link EnergyMicro (441000021) (Nodetest)

Figure 3.10. Radio TX Test Start

Underneath the run button and progress bar, you will see four tabs:

Adapter Tab

The Adapter Tab presents a list of all adapters that are currently running the Nodetest application and can be used for testing.



Ready.

Input parameters are valid, you may run the test.

Select the adapter:

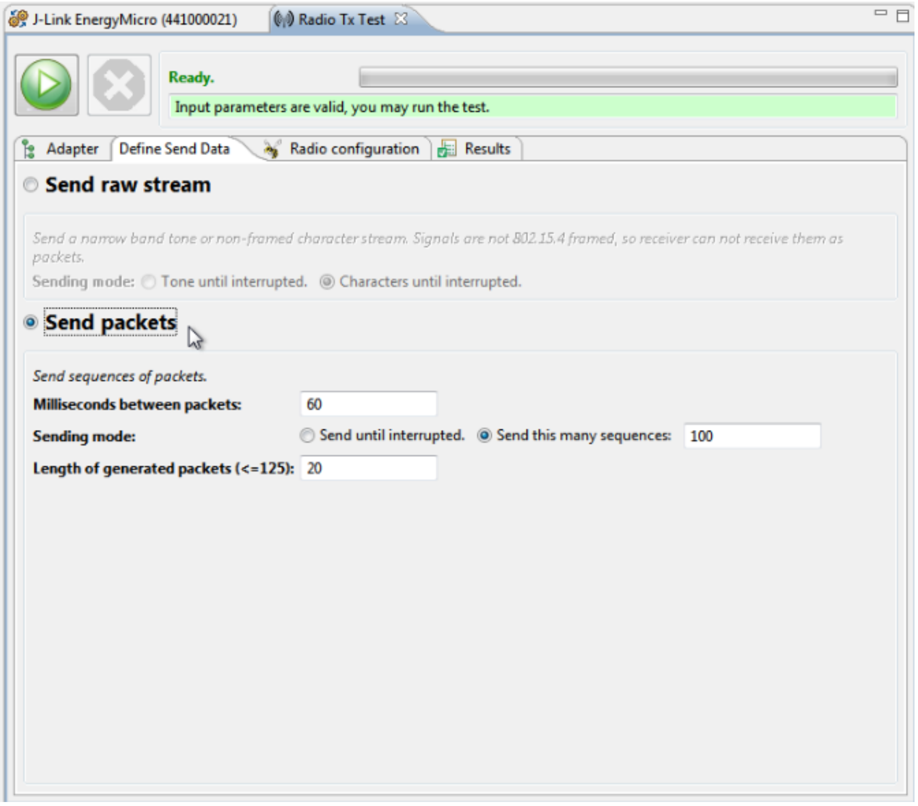
Not seeing adapters? Run discovery or configure adapters.

J-Link EnergyMicro (441000021) (Nodetest)

Figure 3.11. Adapter Tab

Define and Send Data Tab

In this tab you may define the type of test you would like to run. For a Tx test one can send either a raw stream or packets.



Ready.

Input parameters are valid, you may run the test.

○ Send raw stream

Send a narrow band tone or non-framed character stream. Signals are not 802.15.4 framed, so receiver can not receive them as packets.

Sending mode: ○ Tone until interrupted. ● Characters until interrupted.

● Send packets

Send sequences of packets.

Milliseconds between packets: 60

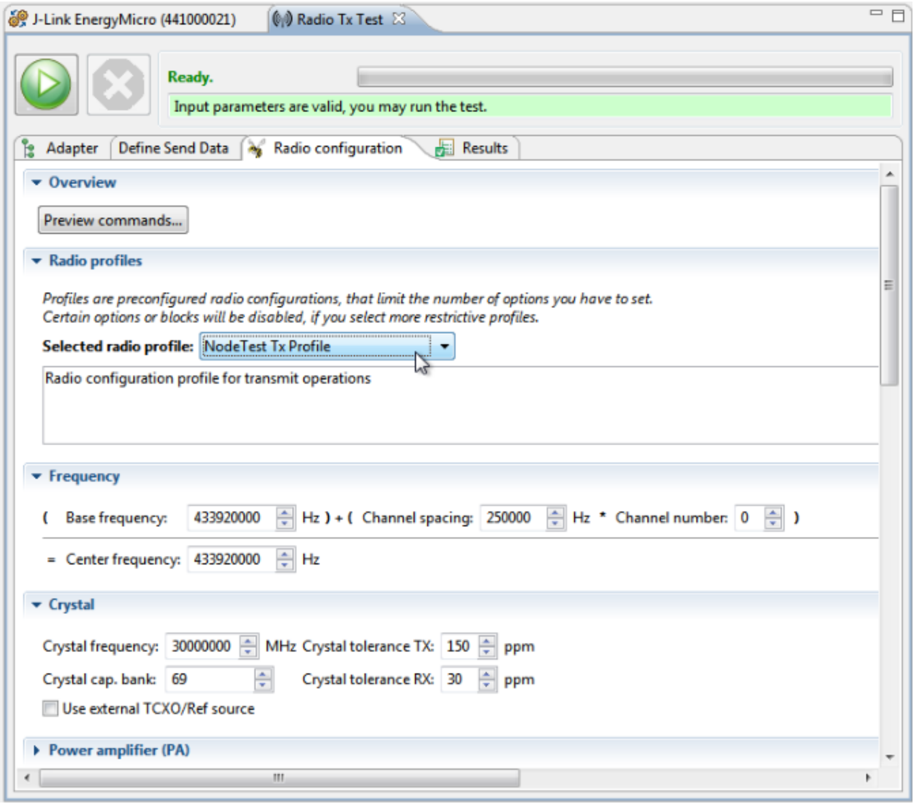
Sending mode: ○ Send until interrupted. ● Send this many sequences: 100

Length of generated packets (<=125): 20

Figure 3.12. Define and Send Data Tab

Radio Configuration Tab

The radio configuration tab sets the parameters for the radio that will be configured before the test is run. This includes settings such as the frequency, crystal, power amplifier, etc.



Ready.

Input parameters are valid, you may run the test.

Profiles are preconfigured radio configurations, that limit the number of options you have to set.
Certain options or blocks will be disabled, if you select more restrictive profiles.

Selected radio profile: NodeTest Tx Profile

Radio configuration profile for transmit operations

(Base frequency: 433920000 Hz) + (Channel spacing: 250000 Hz * Channel number: 0)

= Center frequency: 433920000 Hz

Crystal frequency: 30000000 MHz Crystal tolerance TX: 150 ppm

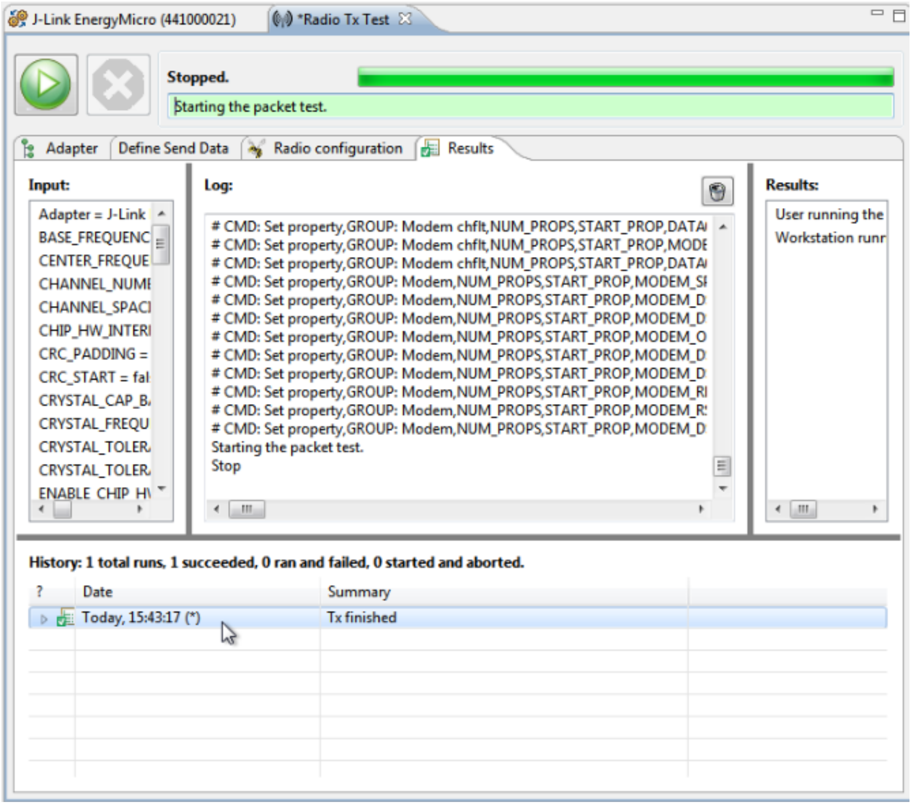
Crystal cap. bank: 69 Crystal tolerance RX: 30 ppm

□ Use external TCXO/Ref source

Figure 3.13. Radio Configuration Tab

Results Tab

The results tab displays a console view of the results for your test, including the output from the radio configuration process that precedes the test.



Stopped.

Starting the packet test.

```
Adapter = J-Link
BASE_FREQUENC
CENTER_FREQUE
CHANNEL_NUMI
CHANNEL_SPACI
CHIP_HW_INTERI
CRC_PADDING =
CRC_START = fal
CRYSTAL_CAP_B.
CRYSTAL_FREQU
CRYSTAL_TOLER.
CRYSTAL_TOLER.
ENABLE_CHIP_H\
```

```
# CMD: Set property,GROUP: Modem chflt,NUM_PROPS,START_PROP,DATA
# CMD: Set property,GROUP: Modem chflt,NUM_PROPS,START_PROP,MODE
# CMD: Set property,GROUP: Modem chflt,NUM_PROPS,START_PROP,DATA
# CMD: Set property,GROUP: Modem,NUM_PROPS,START_PROP,MODEM_SI
# CMD: Set property,GROUP: Modem,NUM_PROPS,START_PROP,MODEM_D
# CMD: Set property,GROUP: Modem,NUM_PROPS,START_PROP,MODEM_D
# CMD: Set property,GROUP: Modem,NUM_PROPS,START_PROP,MODEM_O
# CMD: Set property,GROUP: Modem,NUM_PROPS,START_PROP,MODEM_D
# CMD: Set property,GROUP: Modem,NUM_PROPS,START_PROP,MODEM_D
# CMD: Set property,GROUP: Modem,NUM_PROPS,START_PROP,MODEM_RI
# CMD: Set property,GROUP: Modem,NUM_PROPS,START_PROP,MODEM_R!
# CMD: Set property,GROUP: Modem,NUM_PROPS,START_PROP,MODEM_D
Starting the packet test.
Stop
```

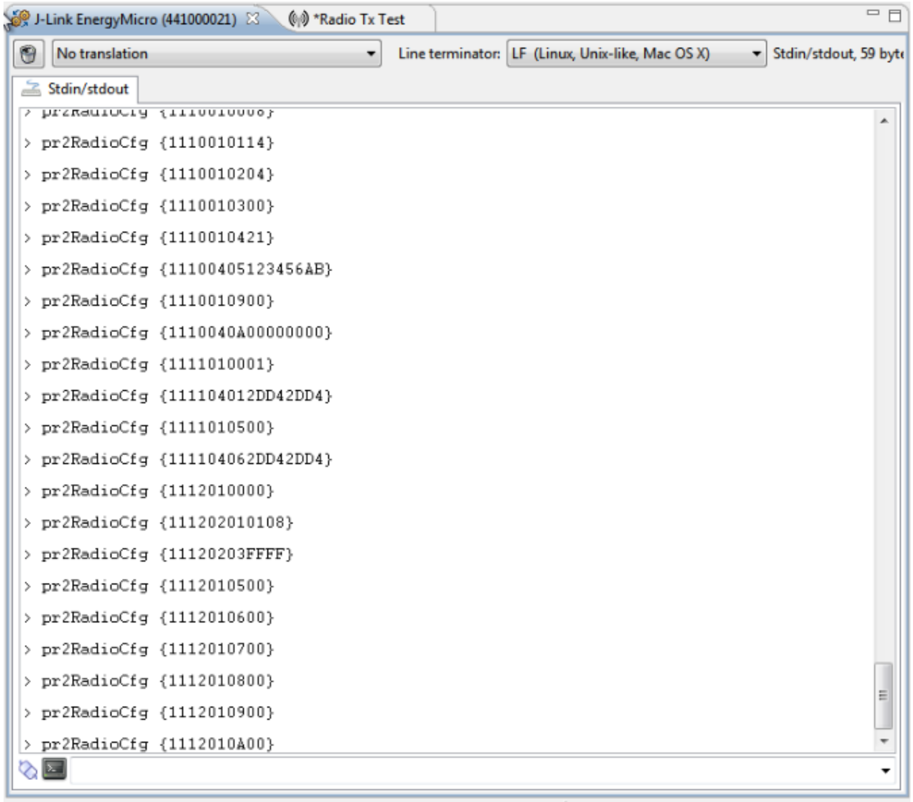
User running the
Workstation runr

History: 1 total runs, 1 succeeded, 0 ran and failed, 0 started and aborted.

?	Date	Summary
	Today, 15:43:17 (*)	Tx finished

Figure 3.14. Results Tab

You may also see the individual commands sent down to the device during the radio configuration process. To do this, re-open the Console view:



No translation

Line terminator: LF (Linux, Unix-like, Mac OS X)

Stdin/stdout, 59 byt

```
> pr2RadioCfg {1110010008}
> pr2RadioCfg {1110010114}
> pr2RadioCfg {1110010204}
> pr2RadioCfg {1110010300}
> pr2RadioCfg {1110010421}
> pr2RadioCfg {11100405123456AB}
> pr2RadioCfg {1110010900}
> pr2RadioCfg {1110040A00000000}
> pr2RadioCfg {1111010001}
> pr2RadioCfg {111104012DD42DD4}
> pr2RadioCfg {1111010500}
> pr2RadioCfg {111104062DD42DD4}
> pr2RadioCfg {1112010000}
> pr2RadioCfg {111202010108}
> pr2RadioCfg {11120203FFFF}
> pr2RadioCfg {1112010500}
> pr2RadioCfg {1112010600}
> pr2RadioCfg {1112010700}
> pr2RadioCfg {1112010800}
> pr2RadioCfg {1112010900}
> pr2RadioCfg {1112010A00}
```

Figure 3.15. Console View

4. The Simple RTx Sample Application

Simplicity Studio comes bundled with an SDK for the EZR32 as part of the EFM packaged sample applications. One of the sample applications, “Simple RTx” can be used to test and work with the radio on the EZR32.

To access the samples available for the EZR32, first select your USB connected device on the left-hand side under discovered devices in the Simplicity Perspective. With the device selected, click on the [**Software Examples**] tile.

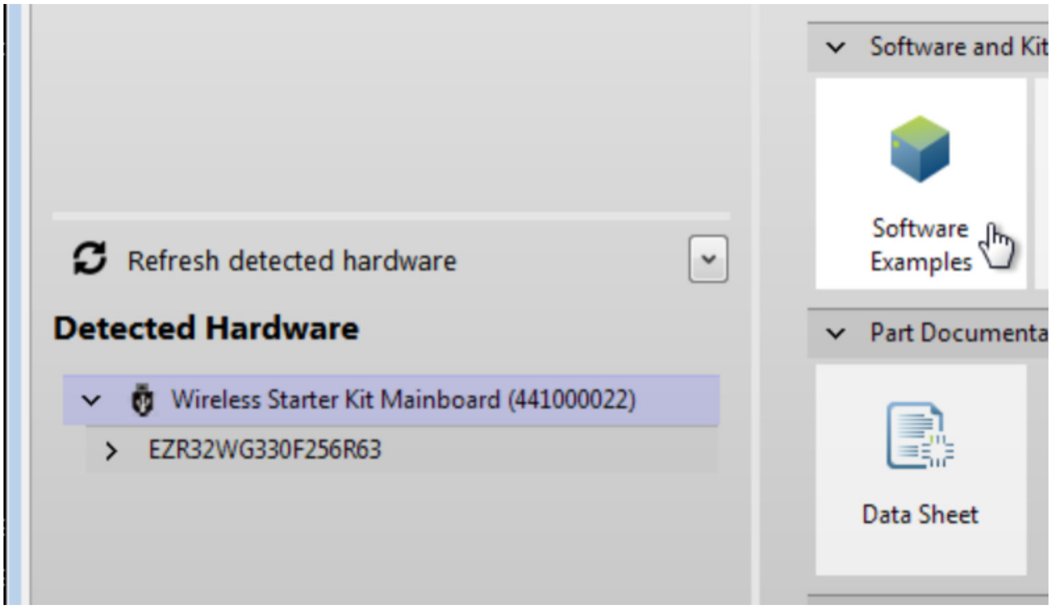
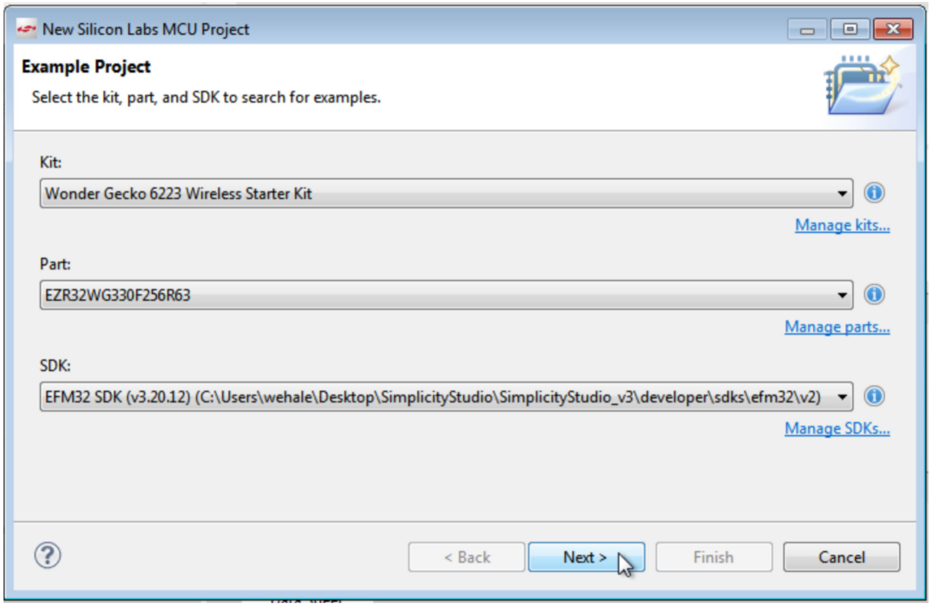


Figure 4.1. Software Example Tiles

This will launch the Example Project dialog window. If you have downloaded the EFM32 SDK, you should see it as an option in the Example Project window.

The Example Project dialog allows you to find the available sample applications for your connected part.



Example Project
Select the kit, part, and SDK to search for examples.

Kit: Wonder Gecko 6223 Wireless Starter Kit

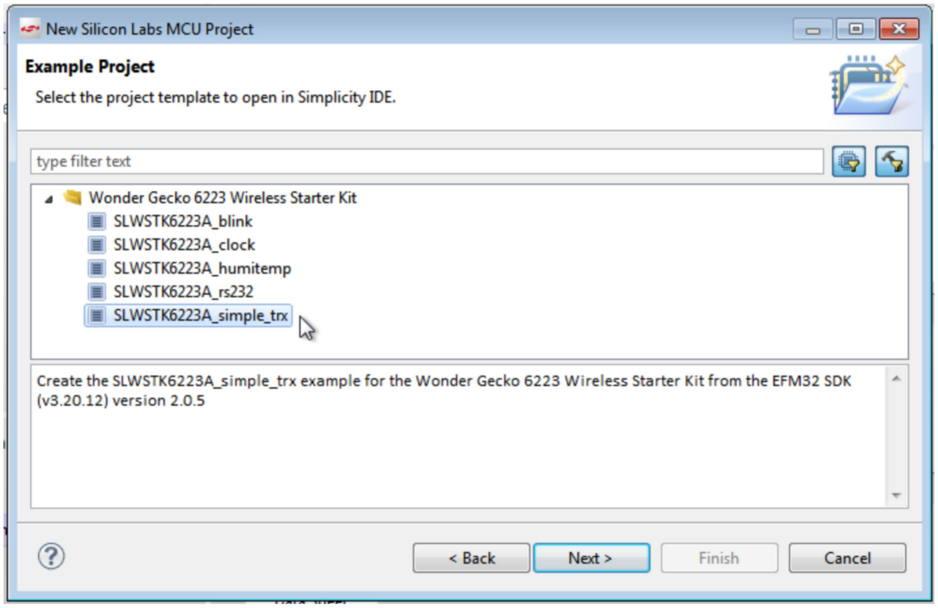
Part: EZR32WG330F256R63

SDK: EFM32 SDK (v3.20.12) (C:\Users\wehale\Desktop\SimplicityStudio\SimplicityStudio_v3\developer\sdks\efm32\v2)

Figure 4.2. Example Project Window

Note: Make sure that you have a Kit selected based on the type of EZR32 you are using. For instance if you are using a Wonder Gecko EZR32 you will want to select one of the Wonder Gecko Wireless starter kits. The sample applications are associated with a specific kit, so a real kit must be selected in order for the sample applications to appear as options.

Click the [**Next**] button. In the next screen you should see all of the sample applications available for your part. Choose the [**simple_rtx**] sample and click [**Next**].



New Silicon Labs MCU Project

Example Project

Select the project template to open in Simplicity IDE.

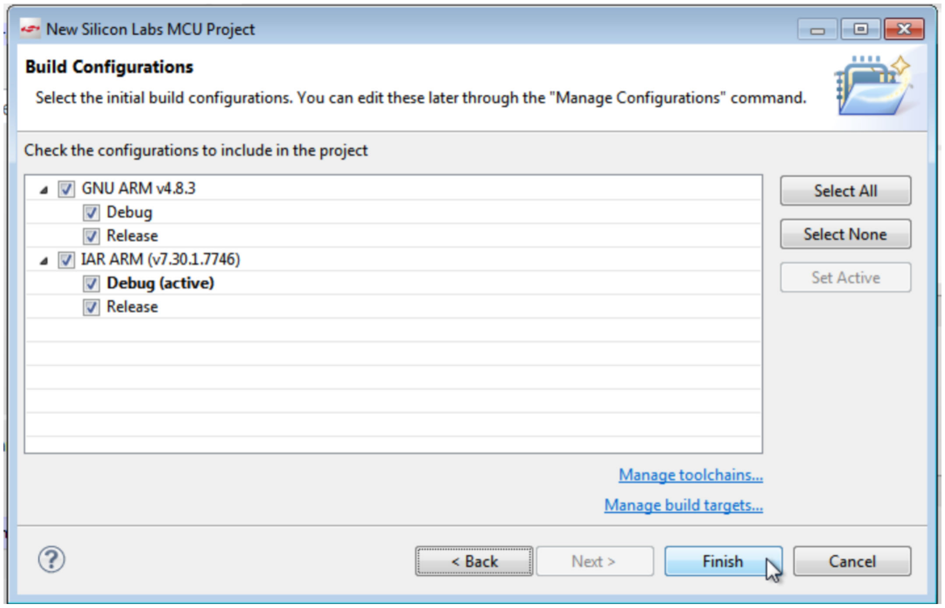
type filter text

- Wonder Gecko 6223 Wireless Starter Kit
  - SLWSTK6223A_blink
  - SLWSTK6223A_clock
  - SLWSTK6223A_humitemp
  - SLWSTK6223A_rs232
  - SLWSTK6223A_simple_trx

Create the SLWSTK6223A_simple_trx example for the Wonder Gecko 6223 Wireless Starter Kit from the EFM32 SDK (v3.20.12) version 2.0.5

Figure 4.3. Select Simple RTx Sample Application

The next screen will allow you to select a name for your application. Choose the name you want and select [**Next**] again. The next screen allows you to configure your Build Configurations for this project. For the Simple Rx/Tx sample you have the option of building with IAR (if you have IAR installed) or GCC. Select all the build configuration options you want and click [**Finish**].



New Silicon Labs MCU Project

Build Configurations

Select the initial build configurations. You can edit these later through the "Manage Configurations" command.

Check the configurations to include in the project

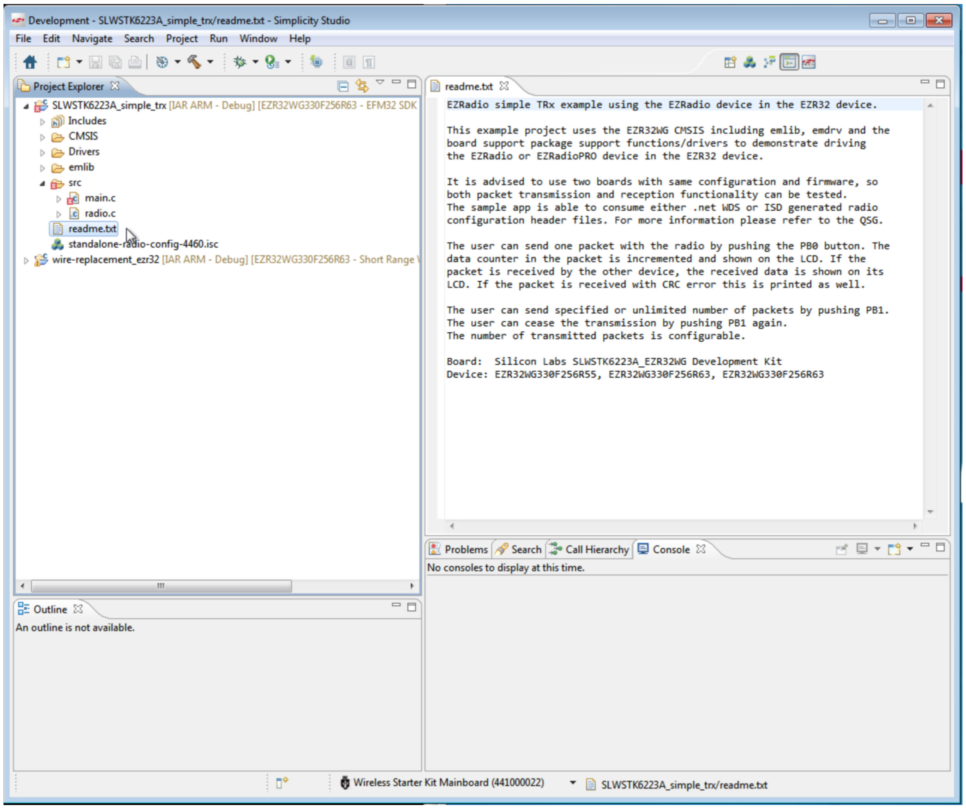
- ☑ GNU ARM v4.8.3
  - ☑ Debug
  - ☑ Release
- ☑ IAR ARM (v7.30.1.7746)
  - ☑ **Debug (active)**
  - ☑ Release

Figure 4.4. Build Configurations

The Simple Rx/Tx sample is developed in the Development Perspective. Clicking on the [**Finish**] button will automatically launch the Development Perspective in which you can edit and build your application. For more information on how to build, load, and interact with your application, please see the “Building Your Sample Application” section.

For more information on how to use the Simple Rx/Tx sample application, please see the `readme.txt` included with the sample and available in your generated application.

All sample applications include a `readme.txt` file that explains the sample and its operation.



Development - SLWSTK6223A_simple_trx/readme.txt - Simplicity Studio
File Edit Navigate Search Project Run Window Help
Project Explorer
SLWSTK6223A_simple_trx [IAR ARM - Debug] [EZR32WG330F256R63 - EFM32 SDK
Includes
CMSIS
Drivers
emlib
src
main.c
radio.c
readme.txt
standalone-radio-config-4460.isc
wire-replacement_ezr32 [IAR ARM - Debug] [EZR32WG330F256R63 - Short Range \
readme.txt
Problems
Search
Call Hierarchy
Console
Outline
Wireless Starter Kit Mainboard (441000022)
SLWSTK6223A_simple_trx/readme.txt

```
EZRadio simple TRx example using the EZRadio device in the EZR32 device.

This example project uses the EZR32WG CMSIS including emlib, emdrv and the
board support package support functions/drivers to demonstrate driving
the EZRadio or EZRadioPRO device in the EZR32 device.

It is advised to use two boards with same configuration and firmware, so
both packet transmission and reception functionality can be tested.
The sample app is able to consume either .net WDS or ISD generated radio
configuration header files. For more information please refer to the QSG.

The user can send one packet with the radio by pushing the PB0 button. The
data counter in the packet is incremented and shown on the LCD. If the
packet is received by the other device, the received data is shown on its
LCD. If the packet is received with CRC error this is printed as well.

The user can send specified or unlimited number of packets by pushing PB1.
The user can cease the transmission by pushing PB1 again.
The number of transmitted packets is configurable.

Board:  Silicon Labs SLWSTK6223A_EZR32WG Development Kit
Device: EZR32WG330F256R55, EZR32WG330F256R63, EZR32WG330F256R63
```

No consoles to display at this time.

An outline is not available.

Figure 4.5. Readme.txt File

For more information on how to load and operate the Simple Rx/Tx sample application, continue to the “Building Your Sample Application” and “Loading Your Sample Application” sections of this Quick Start Guide.

5. Building Your Sample Application

Using Simplicity Studio to Build Your Sample Application

Once your application is generated, you have the ability to compile it using Simplicity Studio. In order to compile your application, simply click on the [**build**] icon in the top tool bar.

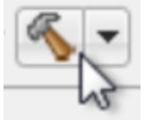
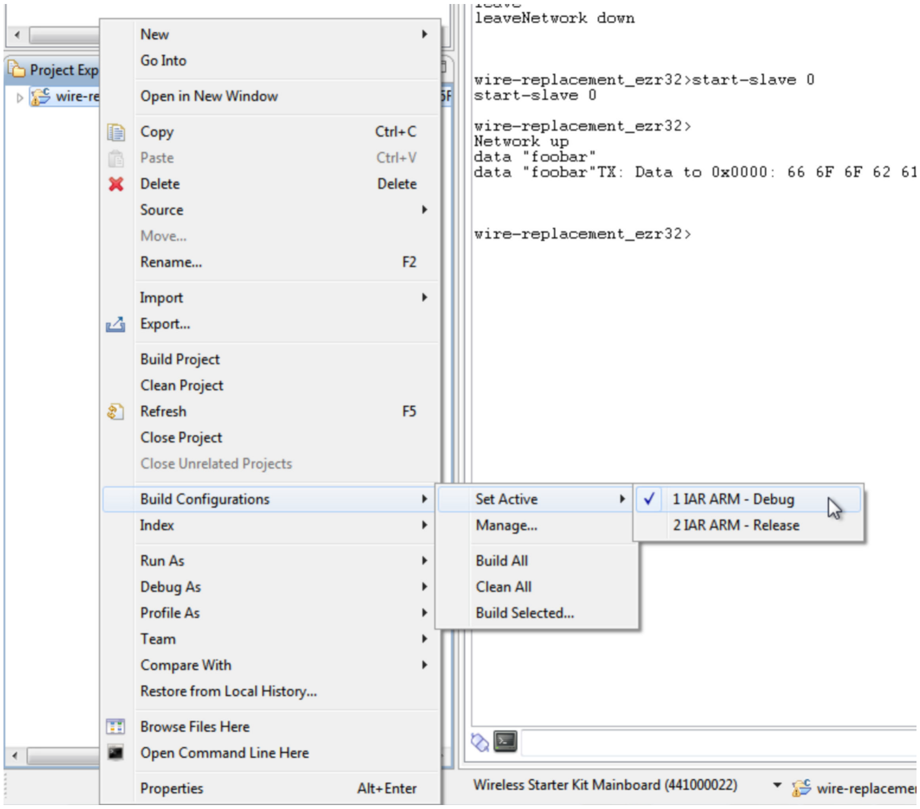


Figure 5.1. Build Button

Your sample application will compile based on its build configuration. You may change the build configuration at any time in the Project Explorer View by right clicking on the [**project**] and going to [**Build Configurations > Set Active**]

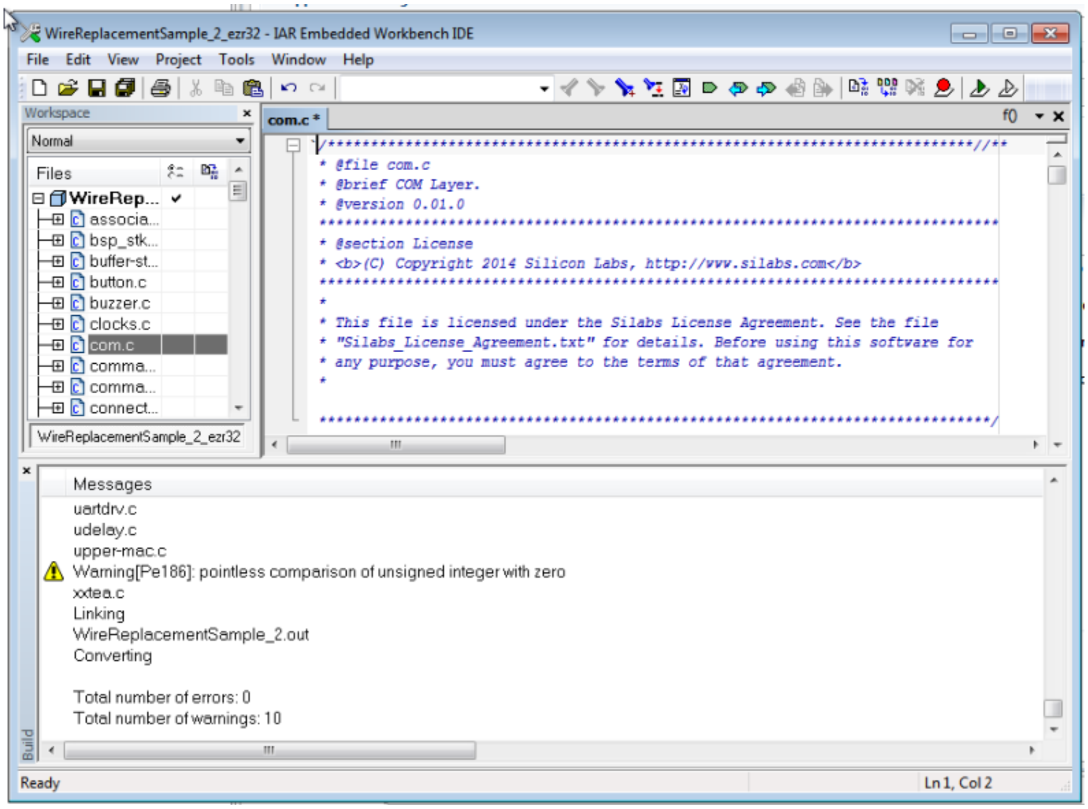


```
leaveNetwork down


wire-replacement_ezr32>start-slave 0
start-slave 0

wire-replacement_ezr32>
Network up
data "foobar"
data "foobar"TX: Data to 0x0000: 66 6F 6F 62 61


wire-replacement_ezr32>
```

Figure 5.2. Change Build Configuration

Using IAR to Build Your Sample Application

You may also build your application directly in IAR by launching IAR and opening the generated project file inside IAR.



```
/*****************************************************************************//**
 * @file com.c
 * @brief COM Layer.
 * @version 0.01.0
 ******************************************************************************
 * @section License
 * <b>(C) Copyright 2014 Silicon Labs, http://www.silabs.com</b>
 ******************************************************************************
 *
 * This file is licensed under the Silabs License Agreement. See the file
 * "Silabs_License_Agreement.txt" for details. Before using this software for
 * any purpose, you must agree to the terms of that agreement.
 *
 ******************************************************************************/
```

```
Messages
uartdrv.c
udelay.c
upper-mac.c
⚠ Warning[Pe186]: pointless comparison of unsigned integer with zero
xxtea.c
Linking
WireReplacementSample_2.out
Converting

Total number of errors: 0
Total number of warnings: 10
```

Figure 5.3. Build Sample Application

6. Loading Sample Application

Loading Sample Application onto EZR32 with Simplicity Studio

Once your sample application is built, you can load it onto your device using Simplicity Studio. There are several ways to do this. If you have compiled your application in Simplicity Studio, you will notice some images listed under the Binaries menu item in the Project Explorer View. You can flash your image to a device by right-clicking on the image you wish to load in the Project Explorer View and selecting the [**Flash to device**] option.

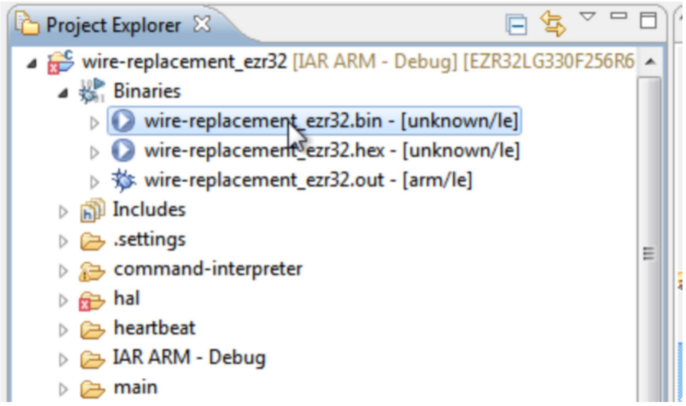
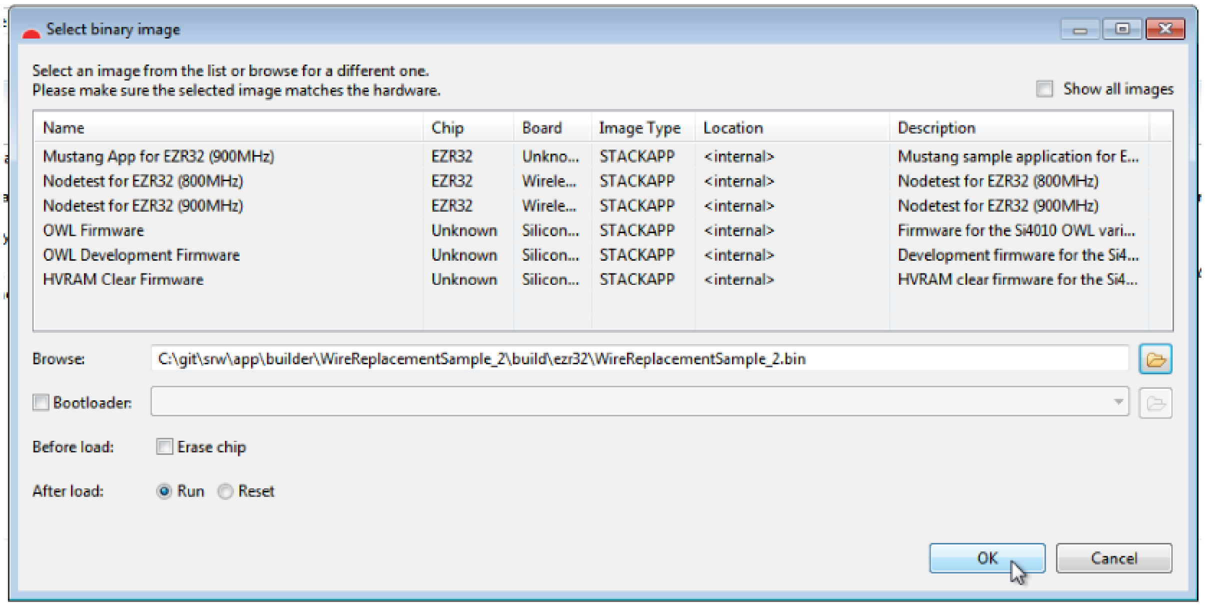

Project Explorer
wire-replacement_ezr32 [IAR ARM - Debug] [EZR32LG330F256R6
Binaries
wire-replacement_ezr32.bin - [unknown/le]
wire-replacement_ezr32.hex - [unknown/le]
wire-replacement_ezr32.out - [arm/le]
Includes
.settings
command-interpreter
hal
heartbeat
IAR ARM - Debug
main

Figure 6.1. Flash Selected Image to Device

You can also upload your image to a device through the Adapters View. To do this go to the Adapters View in Simplicity Studio, which is visible in either the AppBuilder Perspective or in the Network Analyzer Perspective. Right-click on your device of choice, select [**Connect**] and then [**Upload**]. Navigate to the `.bin` or `.hex` image you wish to upload and click [**OK**].



Select an image from the list or browse for a different one.
Please make sure the selected image matches the hardware.

□ Show all images

Name	Chip	Board	Image Type	Location	Description
Mustang App for EZR32 (900MHz)	EZR32	Unkno...	STACKAPP	<internal>	Mustang sample application for E...
Nodetest for EZR32 (800MHz)	EZR32	Wirele...	STACKAPP	<internal>	Nodetest for EZR32 (800MHz)
Nodetest for EZR32 (900MHz)	EZR32	Wirele...	STACKAPP	<internal>	Nodetest for EZR32 (900MHz)
OWL Firmware	Unknown	Silicon...	STACKAPP	<internal>	Firmware for the Si4010 OWL vari...
OWL Development Firmware	Unknown	Silicon...	STACKAPP	<internal>	Development firmware for the Si4...
HVRAM Clear Firmware	Unknown	Silicon...	STACKAPP	<internal>	HVRAM clear firmware for the Si4...

Browse: C:\git\srw\app\builder\WireReplacementSample_2\build\ezr32\WireReplacementSample_2.bin

□ Bootloader:

Before load: □ Erase chip

After load: ● Run ○ Reset

Figure 6.2. Uploading `.bin` or `.hex` Image

Note: Please make sure that the power switch on your WSTK board is set in the “AEM” position when attempting to flash your device.

Simpilcity Studio

One-click access to MCU tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

www.silabs.com/simplicity

MCU Portfolio
www.silabs.com/mcu

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support and Community
community.silabs.com

Disclaimer
Silicon Laboratories intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Silicon Laboratories products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications. Application examples described herein are for illustrative purposes only. Silicon Laboratories reserves the right to make changes without further notice and limitation to product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Silicon Laboratories shall have no liability for the consequences of use of the information supplied herein. This document does not imply or express copyright licenses granted hereunder to design or fabricate any integrated circuits. The products must not be used within any Life Support System without the specific written consent of Silicon Laboratories. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Silicon Laboratories products are generally not intended for military applications. Silicon Laboratories products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons.

Trademark Information
Silicon Laboratories Inc., Silicon Laboratories, Silicon Labs, SiLabs and the Silicon Labs logo, CMEMS®, EFM, EFM32, EFR, Energy Micro, Energy Micro logo and combinations thereof, "the world's most energy friendly microcontrollers", Ember®, EZLink®, EZMac®, EZRadio®, EZRadioPRO®, DSPLL®, ISOmodem ®, Precision32®, ProSLIC®, SiPHY®, USBXpress® and others are trademarks or registered trademarks of Silicon Laboratories Inc. ARM, CORTEX, Cortex-M3 and THUMB are trademarks or registered trademarks of ARM Holdings. Keil is a registered trademark of ARM Limited. All other products or brand names mentioned herein are trademarks of their respective holders.

SILICON LABS

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

http://www.silabs.com